MOBILE ACADEMY

Maciej Oczko

@MaciejOczko

maciejoczko.pl

#TDDCraftConf

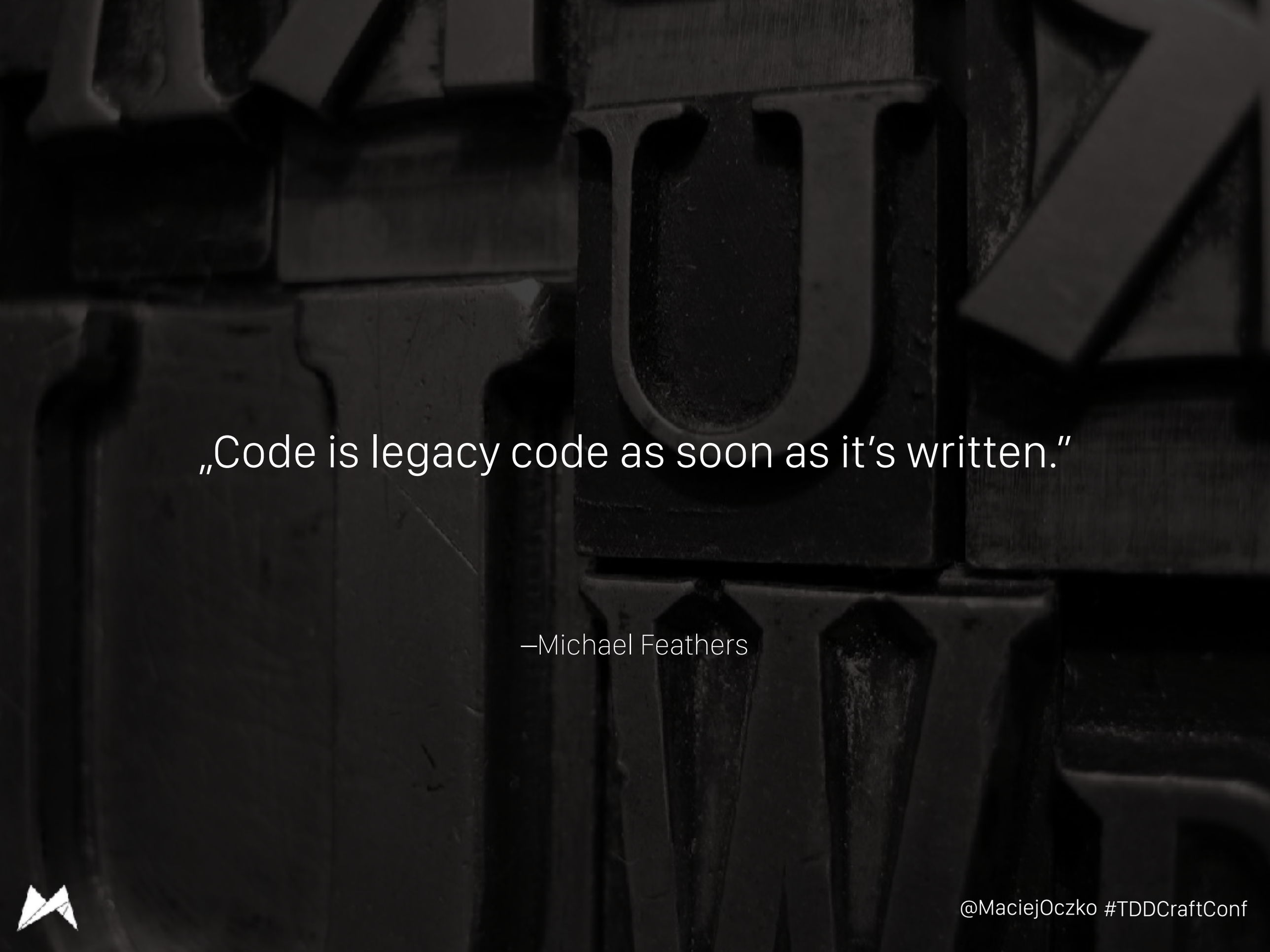
Working with Legacy Code

What is Legacy Code?

How to deal with it?

Refactor tips

What is Legacy Code?


„Code is legacy code as soon as it’s written.”
—Michael Feathers

Legacy code:

Code without tests

Legacy code

- inherited
- poorly designed
- too complicated
- not readable

Hands on #1

Pair Programming

Hands on #1

Pair Programming

RED  A

GREEN  B

REFACTOR  A

Hands on #1

1. origin/task/poll/task-1
2. Look at `PollViewController viewWillAppear` method.
3. Test, if `rightBarButtonItem` is set correctly depending on `PollManager isPollAlreadySent` property value.

Hands on #1

Test, if `rightBarButtonItem` is set correctly depending on `PollManager isPollAlreadySent` property value.

1. Create PollViewControllerSpec.swift file
2. Remember about adding it to test target
3. To create spec from live template use: qspec
4. Useful templates: qdesc, qcon, qit, qbef, qaft
5. Expectation template: nex

Hands on #1

`origin/task/poll/solution-1`

What is Legacy Code?

How to deal with it?

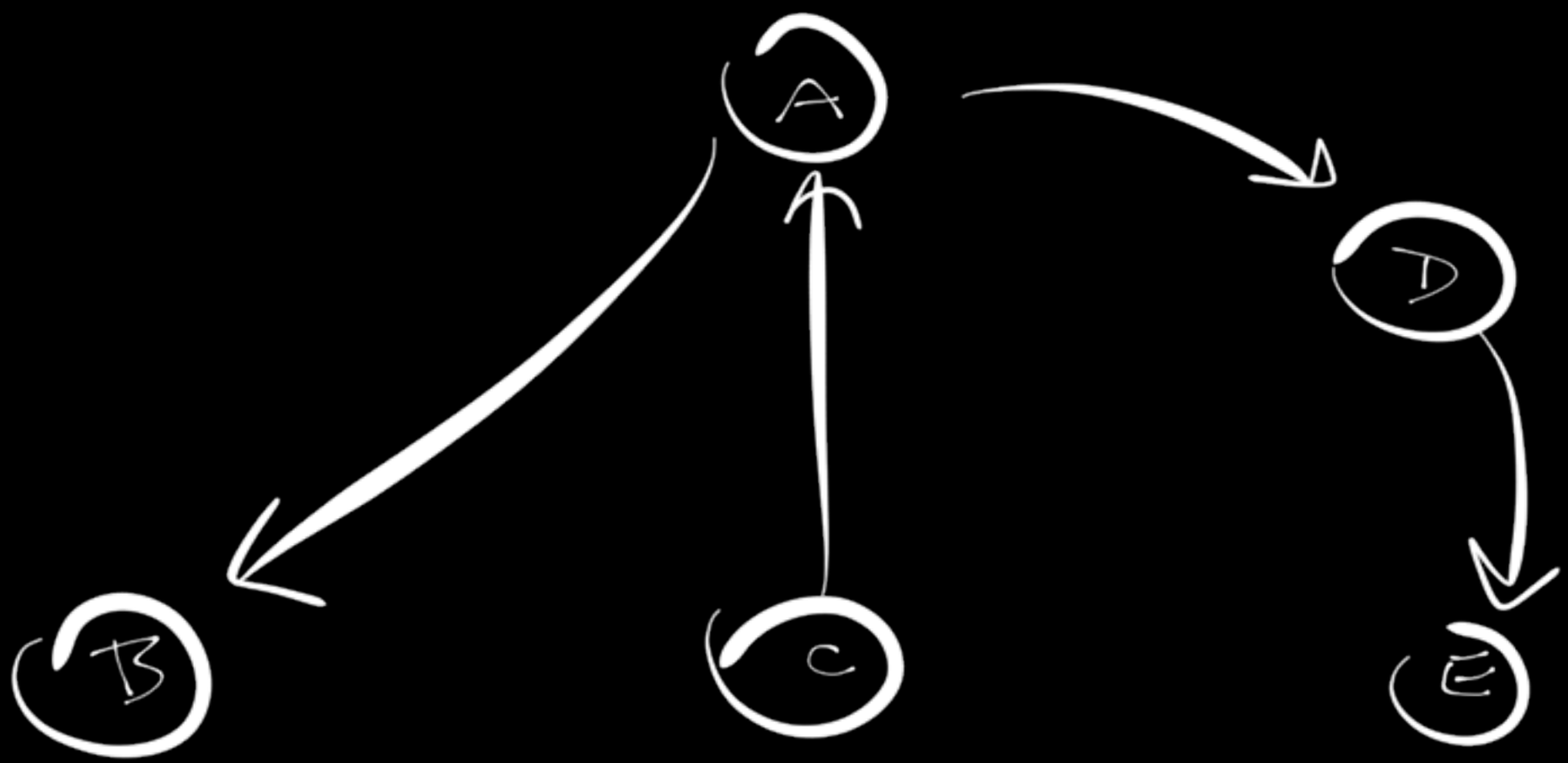
Refactor tips

How to deal with it?

Approach

- identify change points
- find an infection point
- cover the inflection point
- make changes
- refactor the covered code

inflection point

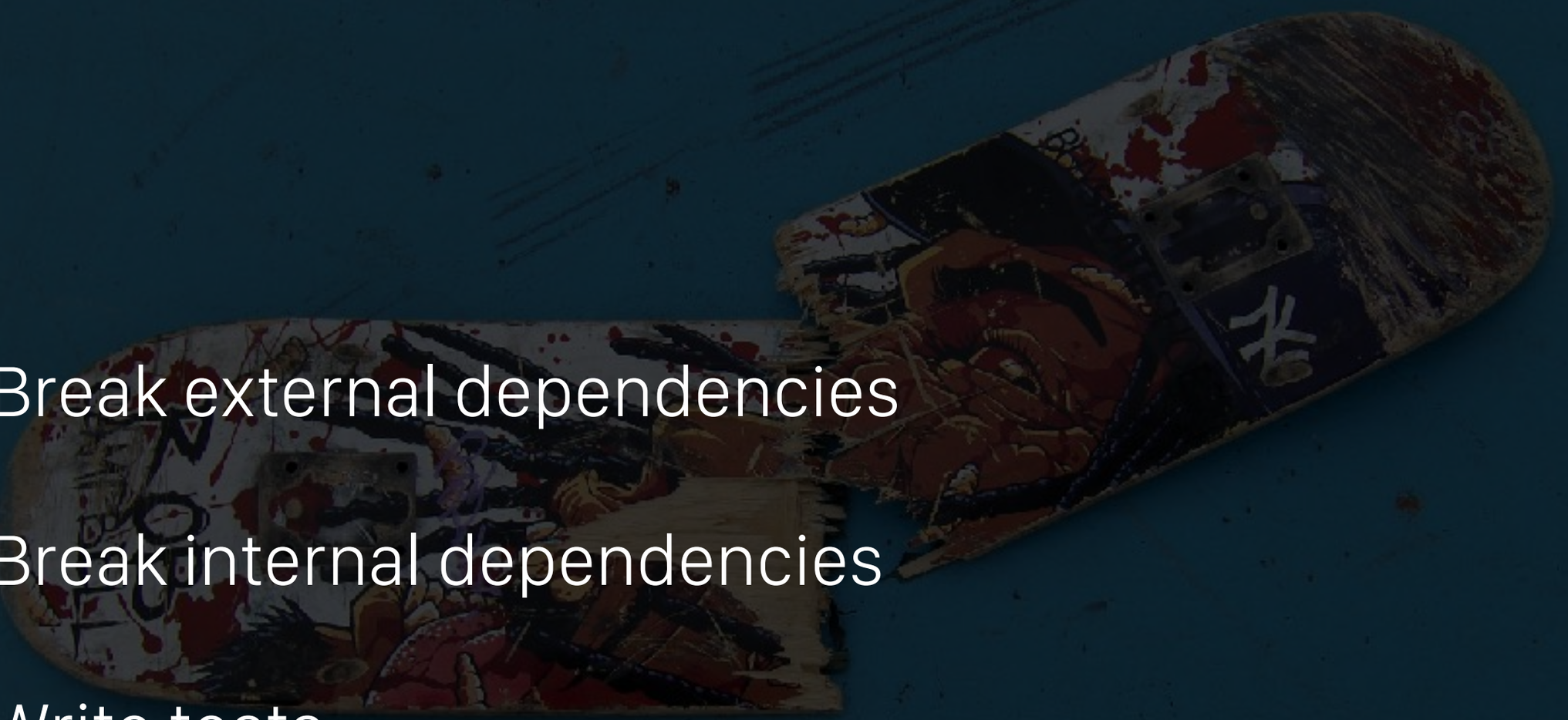


Single responsibility principle

Open-closed

principle

Covering inflection point



- Break external dependencies
- Break internal dependencies
- Write tests

Breaking **external** dependencies is about moving...

From this

```
func calculatePrice() -> Int {
  // ...
  let stockAnalyzer = StockAnalyzer()
  // ...
}
```

To this

```
init(with analyzer: StockAnalyzer) {
  self.analyzer = analyzer
}
```

```
func calculatePrice() -> Int {
  // ...
  let analysis = analyzer.createReport()
  // ...
}
```

Breaking **internal** dependencies is about moving...

From this

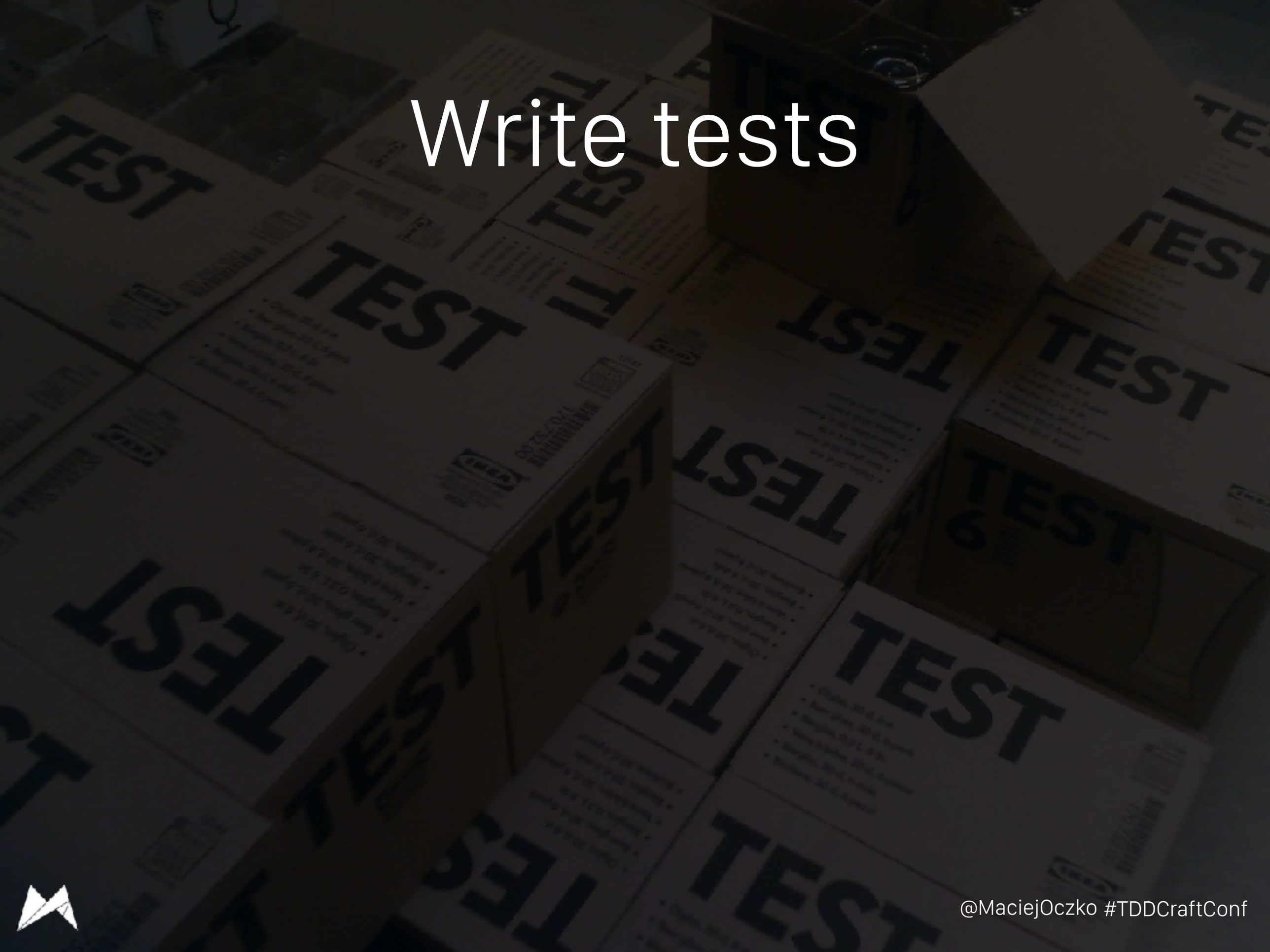
```
func calculateSize() -> CGSize {
  // ...
  let screen = UIScreen.main
  // ...
}
```

To this

```
func calculateSize() -> CGSize {
  // ...
  let screen = self.screen()
  // ...
}

func screen() -> UIScreen {
  return UIScreen.main
}
```

Write tests



Make

changes

Hands on #2

1. origin/task/poll/task-2
2. Look at `PollViewController` validation methods.
3. Extract text validation logic, test it and test the view controller (follow the hints).

Hands on #2

`origin/task/poll/solution-2`

What is Legacy Code?

How to deal with it?

Refactor tips

Refactor tips

Too wide class responsibility

```
class MyClass {
  let composer: MessagesComposer
  let reader: AlbumReader
  let parser: FeedParser
  // ...
}
```

Duplication

Not readable code

```
char*_ = "'"/*";
#include <stdio.h>
#define m 21
#define o(l, k) for(l=0; l<k; l++)
#define n(k) o(T, k)

int E,L,O,R,G[42][m],h[2][42][m],g[3][8],c
[42][42][2],f[42]; char d[42]; void v( int
b,int a,int j)( printf("\33[%d;%df\33[4%d"
"m  ",a,b,j); ) void u()( int T,e; n(42)o(
e,m)if(h[0][T][e]-h[1][T][e])( v(e+4+e,T+2
,h[0][T][e]+1?h[0][T][e]:0); h[1][T][e]=h[
0][T][e]; ) fflush(stdout); ) void q(int l
                ,int k,int p)(
                int T,e,a; L=0
                ; O=1; while(O
                )( n(4&&L)( e=
                k+c[l] [T][0];
                h[0][L-1+c[l][
                T][l]][p?20-e:
e]=-1; ) n(4)[                                  e=k+c[l][T][0]; a=L+c[l][T][
l]-1; if(a==42                                  || h[0][a][p?20-e:e]+1)( O=0
; ) ) n(4)( e=                                  k+c[l][T][0]; h[0][L + c[l][
T][l]][p?20-e:                                  e]=g[l][f[p?19+l:l]]; ) L++;
u(); ) n(42) [                                  o(e,m)if(h[0][T][e]<0)break;
o(a, m&&e==m){                                  for(L=T; L; L--) ( h[0][L][a
]=h[0][L-1] [a                                  ]; ) h[0][0][a]=-1; ) ) u();
)int main(){ int T,e,t,r,i,s                ,D,V,K; printf("\33[2J\33[?25l"); n(8)g[i=
1][T]=7-T; R--; n(42) o(e,m)                G[T][e]--; while(fgets(d,42,stdin)) { r=++
R; n(17){ e=d[T]-48; d[T]=0;                if ((e&7)==e) { g[0][e] ++; G[R][T+2]=e; }
} } n(8)if(g[0][7-T]){ t=g[i                ][O]; g[i][O++]=g[i][T]; g[i][T]=t; ) n(8)
g[2][g[1][T]]=T; n(R+1)o(e,m                )if(G[T][e]+1) G[T][e]=g[2][G[T][e]]; n(19
)o(t,2){ f[T+t+T]=(T["+%W,4"                "5>GP9$5-.#C?NX"]-35)>>t*3&7; o(e,4){ c[T]
[e][t]=("5'<$=$8)Ih$=h9i3'9"                "t=)83)14(99(g9>##>4(" [T-t+T]-35)>>e*2&3;
} } n(15) { s=T>9?m:(T&3)-3?15:36;o(e,s)o(t,2)c[T+19][e][t]="6*6,8*6.608.6264826668\
865::{+;0(6+6-6/8,61638055678469.;88)}()3(6,8*6.608.6264826668865:+:4)-*6-6/616365,\
-0?15090.5;,89,81+,(023090/:40(8-7751)2)05;095(855(+*8)+;4^^+4(((0.608.626482000880\
5:-;4+4)0(8)6/61638065679469.;88)-4,4*8+4(((60(/6264826668865:+;4-6|6365676993-9:54\
+-14).:./347.+18*):1;-*0-975/)936.+:4*,80987(887(0(*)4.*'"/4,4*8+4(((6264826668865:\
+;4/4-4+8-4)0[8)6365678489.;88)1/(6*6,6.60626466686:8)8-8*818.8582/9863(+;/""*6,6.6\
0626466686:4(8)8-8*818.8582/9863(+;/,6.60626466686:8-818.8582/9864*4+4(0())+;/,6062\
6466686:8/8380/7844,4-4*4+4(0())60+;/0626466686:818582/0864.4/4,4-4*4+4(0())+;' [o+E
+e-t]-40; E+=s+s; ) n(45)( if(T>i) ( v(2,T,7); v(46,T,7); ) v(2+T,44,7); ) T=0; o(e,
42)o(t,m)h[T][e][t]--; while(R+i) { s = D=0; if (r-R) { n(19) if (G[R+i][T]+i) V=T/2
; else if(G[R][T]+i) s++; if(s) ( if(V>4)( V=9-V; D++; ) V+=29; n(20) q(c[V][T][0],c
[V][T][i],D); ) } n(19) if((L=G[R][T])+i) ( O=T-L; e=O>9; t=e?18-O :O; o(K,((t&3)-3?
16:37))( if(K)( L=c[t+19][K-i][0]; O=c[t+19][K-i][i] ; ) q(L,O,K && e); ) ) if(s) q(
c[V][20][0], c[V][20][i], D); R--; } printf("\33[47;1f\33[?25h\33[40m"); return 0; }
```

Too many method arguments

```
func showPresentation(
        _ presentation: Presentation,
        shuffled: Bool,
        enablingInteraction: Bool,
        type: PresentationType,
        options: [String: Any]
    ) {
    // ...
}
```

Composition over inheritance

Singleton avoidance

Dependencies isolation

Thanks!

@MaciejOczko

github.com/literator

#TDDCraftConf

MOBILE ACADEMY